7-2014

Getting it right the first time: Verification of Behavior-based Multirobot Missions

Damian M. Lyons
Fordham University

Ronald C. Arkin
Georgia Institute of Technology

Shu Jiang
Georgia Institute of Technology

Dagon Harrington
Fordham University

Matthew O'Brien
Georgia Institute of Technology

Abstract: Position Paper
Getting it right the first time: Verification of Behavior-based Multirobot Missions

Damian Lyons[1], Ronald Arkin[2], Shu Jiang[2], Dagan Harrington[1] and Matthew O'Brien[2]
[1]Dept. Of Computer & Information Science, Fordham University, Bronx NY 10458
[2]School of Interactive Computing, Georgia Institute of Technology, Atlanta GA 30332
{dlyons,dharrington5}@fordham.edu, {arkin, sjiang, mjobrien}@cc.gatech.edu

I. INTRODUCTION

In research being conducted for the Defense Threat Reduction Agency (DTRA), we are concerned with robot missions that may only have a single opportunity for successful completion, with serious consequences if the mission is not completed properly. In particular we are investigating missions for Counter-Weapons of Mass Destruction (C-WMD) operations, which require discovery of a WMD within a structure and then either neutralizing it or reporting its location and existence to the command authority. Typical scenarios consist of situations where the environment may be poorly characterized in advance in terms of spatial layout, and have time-critical performance requirements. It is our goal to provide reliable performance guarantees for whether or not the mission as specified may be successfully completed under these circumstances, and towards that end we have developed a set of specialized software tools to provide guidance to an operator/commander prior to deployment of a robot tasked with such a mission.

II. VERIFICATION FOR BEHAVIOR-BASED MISSIONS

Automatic verification of software is a very desirable functionality in any application where software failure can incur heavy penalties [6]. While we know that a completely general solution is ruled out by the undecidability of the halting problem, much research has been conducted on restricted instances of the problem. Model checking is a collection of techniques that conduct an exhaustive exploration of the state-space of a program [3] to determine whether the program satisfies a temporal logic constraint on its behavior.

More recently, some researchers have effectively leveraged model-checking techniques to address the *correct-by-construction* robot control problem [2][9]. A solution to the correct-by-construction problem takes as input a temporal logic description of the desired behavior of the robot controller and then fabricates a controller guaranteed to abide by this description.

Our problem differs from the correct-by-construction problem, and is similar to the general-purpose software verification problem, in that our input is mission software designed using the *MissionLab* toolkit [16], and our objective is to verify that this software abides by a performance constraint.

It is similar to the correct-by-construction problem in that we require a model of the environment in which the software is to be carried out, something not typically explicit in general-purpose software verification [7].

However, our problem differs from both in needing to efficiently process probabilistic software and environment models, continuous environment characteristics and asynchronous and concurrent environment dynamics. These problem aspects are troublesome for model-checking approaches: One of the biggest contributions to state-space explosion in model-checking is the translation from program to formal model. It is exponential in the number of program variables, and becomes infinite if a variable domain is infinite [3]. After translation, asynchronous concurrent modules are another formidable contributor to complexity, since the concurrent system state space is the Cartesian product of the component spaces.

A. Process-Algebra Approach

For all of these reasons, our approach to the problem focuses on avoiding an explicit state-space representation and especially one in which the number of program variables will introduce exponential complexity [1][8][11]-[15]. We leverage a process-algebra representation to develop a solution in which the program is translated to a set of equations over the program variables, which include random variables with mixture of Gaussian distributions. This translation is strongly based on the structure of behavior-based programs in *MissionLab;* it would be more difficult to do this for arbitrarily structured software systems. We construct solutions to these equations by mapping them to a Dynamic Bayesian Network and applying a filtering algorithm.

B. Performance Guarantees and Environment Models

Using process-algebra as our formal representation for the mission software means that we have the option to also use this, rather than a temporal logic, as the language for the performance guarantee as well as for the description of the environment models. When process-algebra is used for specification [4][10] a major difficulty encountered is specifying proscription (e.g., the safety property that the robot does not collide). Our variation on an implementation bisimulation [5] between the system (mission software and

environment model) and the performance guarantee separates constraints on process ordering from conditions on parameter values, supporting proscription.

We do not propose that *MissionLab* designers build, in detail, their own environment models against which to test the mission. Instead, we propose that a set of standard environment models be constructed a-priori and provided as a library from which robot, sensor and environment features can be selected and composed automatically into an environment model.

The process-algebra we use employs communication ports and port-to-port connections for concurrent modules. This facilitates specifying plug-and-play compatible environment models, since the formal model of the mission software just communicates over a set of ports with any selected environment model. The development of a standard set of environment models is not something we have pursued beyond those we have developed and used in validation.

C. Validation

Because we are verifying probabilistic systems, it is crucial to validate our predicted performance guarantees by carrying out physical robot experimentation. Calibration data is collected on the robots and sensors used in missions, and suitable environment models constructed. We have verified and validated single and multiple waypoint missions, exploration style missions, and multiple robot missions. In each case, we verify a selected performance guarantee for the mission. Because the system is probabilistic, typically representing environment uncertainty, the verification answer is not a binary yes/no, but a probability landscape capturing the system's performance. The mission is validated by carrying out multiple physical runs and collecting performance statistics on real robots. We compare the validation and verification results to evaluate the quality of our verification prediction.

III. FORMAL METHOD

The core of our approach is the process-algebra formal models of mission software, environment and performance guarantee. There are several steps in the translation from *MissionLab* and the verification against the performance guarantee that can be described in detail:

1. The graphical behavior-based CfgEdit program in MissionLab is translated to a process algebra formal representation.
2. The environment model is selected and composed with the mission software to produce a concurrent and communicating system.
3. An expansion theorem in process algebra relates concurrent to sequential composition. The *system period* expansion theorem [12] allows us to transform the concurrent system to a tail-recursive sequential system.
4. One of our key results is the method by which the tail-recursive sequential system is analyzed to produce a set of equations in the program variables that characterize the program, avoiding the combinatorics introduced in going from program to state model in model checking [13].

5. The set of equations is used to build a Dynamic Bayesian Network that evaluates these equations using a probabilistic filtering algorithm [15].
6. An implementation bisimulation relates a performance constraint to a system by considering the system to be a more fully detailed implementation of the performance constraint. The bisimulation is used to derive a goal condition for filtering. If the goal is met, then the performance guarantee is verified. If instead the maximum time is exceeded, then the guarantee is not verified.

IV. VALIDATION

MissionLab is a software tool for designing, simulating, executing and monitoring behavior-based autonomous robot missions. Once a mission has been verified, that same mission can be simulated or executed in a straightforward fashion. We have validated several c-WMD missions to understand the quality of our predictions. The environment model in each case includes motion uncertainty for a Pioneer 3-AT moving in a flat indoor surface. The following can be described in detail:

1. Single and multiple motions of a single robot to a destination in free space [13].
2. Motion of a robot with a hard collision constraint (can collide with the wall) [14].
3. Single robot multiple waypoint missions [15].
4. Single robot exploration missions (searching for a target using a target sensor) [8].
5. Multiple robot waypoint missions (bounding overwatch).
6. Multiple robot missions including obstacle avoidance.

Figure 1 shows an example of the validation/verification comparison, in this case for various completion times and spatial success criterion for a multirobot bounding overwatch mission.
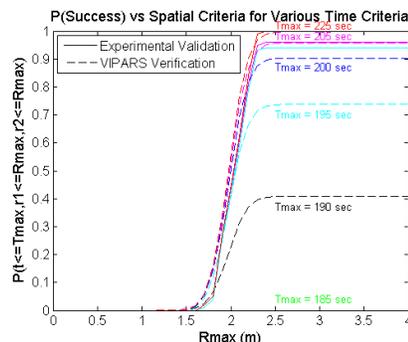

Fig.1: Verification & Validation of Spatial Criterion at various T_{max}

REFERENCES

[1] Arkin, R. C., Lyons, D., Jiang, S., Nirmal, P., &Zafar, M. (2012) Getting it right the first time: Predicted performance guarantees from the analysis of emergent behavior in autonomous and semi-autonomous systems. *Proceedings of SPIE*. Vol. 8387.

[2] Belta, C. (2010) Synthesis of provably-correct control and communication strategies for distributed mobile systems. *ICRA'10 Workshop on Formal Methods for Rob.& Aut.*

[3] Baier, C., and Katoen, J-P., *Introduction to Model Checking*. MIT Press 2008.

[4] Bolognesi, T., and Brinksma,E. (1987) Introduction to the ISO Specification Language LOTOS, *Computer Networks & ISDN Sys,* 14(1), pp. 25-59.

[5] De Nicola, R. (1987) Extensional Equivalences for Transition Systems, Acta Informatica, 24:211-237.Shankar, N. (2009) Automated deduction for Verification. *ACM Computing Surveys* 41(4) 20:1-56.

[6] Hinchey M.G., and J.P. Bowen, *High-Integrity System Specification and Design*, FACIT series, Springer-Verlag, London, 1999.

[7] Jhala, R., Majumdar, R., Software Model Checking. *ACM Computing Surveys,* V41 N4, Oct 2009.

[8] Jiang, S., Arkin, R., Lyons, D., Liu, T-M., and Harrington, D. (2013) Performance Guarantees for C-WMD Missions. IEEE*Int. Symp. Safety, Sec.& Res. Rob.*, Linkoping Sweden.

[9] Johnson, B., and Kress-Gazit, H., Probabilistic Analysis of Correctness of High-Level Robot Behavior with Sensor Error, *Robotics Science and Systems*, 2011.

[10] Karaman, S., Rasmussen, S., Kingston, D., Frazzoli, E., Specification and Planning of UAV Missions: A Process Algebra Approach. *2009 American Control Conference*, St Louis MO, June 2009.

[11] Lyons, D., Arkin, R. (2004) Towards Performance Guarantees for Emergent Behavior. *IEEE Int. Conf. on Rob. & Aut.*

[12] Lyons, D., Arkin, R., Nirmal, P and Jiang, S., (2012) Designing Autonomous Robot Missions with Performance Guarantees.. *IEEE/RSJ IROS,* Vilamoura Portugal.

[13] Lyons, D., Arkin, R., Nirmal, P and Jiang, S., Liu, T-L. (2013) A Software Tool for the Design of Critical Robot Missions with Performance Guarantees. *Conf. Sys. Eng. Res. (CSER'13)*.

[14] Lyons, D., Arkin, R., Liu, T-L., Jiang, S., Nirmal, P. (2013a) Verifying Performance for Autonomous Robot Missions with Uncertainty. *IFAC Int. Vehicle Symp,* Gold Coast, Australia.

[15] Lyons, D., Arkin, R., Nirmal, P and Jiang, Liu, T.M., S., Deeb, J. (2013b) Getting It Right The First Time: Robot Mission Guarantees in the Presence of Uncertainty. *IEEE/RSJ IROS,* Tokyo, Japan.

[16] MacKenzie, D., Arkin, R.C., Cameron, R. (1997) Multiagent Mission Specification and Execution. *Aut. Robots* 4(1): 29-52.